مدرس: **دکتر فدایی، دکتر یعقوب‌زاده** طراح: محمد امانلو، امین یوسفی

Markov Decision Process

سوال اول) سوالات مفهومی

(الف) در کلاس یاد گرفتیم که معادلات بلمن می‌توانند برای توصیف بهره‌وری بهینه در MDPها استفاده شوند. به‌عنوان مرجع، این معادله به این صورت بیان می‌شود:

$$V^*(s) = \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^*(s')]$$

در این معادله، ۷ چه نامیده می‌شود؟ چرا ضروری است؟ وقتی ۷ بزرگ‌تر می‌شود چه اتفاقی می‌افتد؟ و اگر کوچک‌تر شود چه تأثیری دارد؟

پاسخ: ۷ به عنوان ضریب تخفیف شناخته می‌شود و معمولاً در بازه [0, 1] است. نیاز است که این ضریب کمتر از ۱ باشد تا تضمین کند الگوریتم‌ها همگرا می‌شوند (و از بی‌نهایت شدن پاداش‌ها در صورت طولانی شدن بازی جلوگیری می‌کند). از نظر شهودی نیز منطقی است که پاداش‌های فوری را نسبت به پاداش‌های آینده ارزشمندتر بدانیم.
اگر ۷ کوچک‌تر باشد، نشان‌دهنده‌ی یک افق دید (Horizon) کوتاه‌تر یا تمرکز کوتاه‌مدت است. اگر ۷ به ۱ نزدیک شود (بدون Discount)، به این معناست که پاداش‌ها در هر لحظه ارزش یکسانی دارند. اما اگر ۷ به صفر نزدیک شود، فقط پاداش‌های فوری ارزشمند می‌شوند.

(ب) تفاوت‌های کلیدی بین الگوریتم‌های ارزش‌یابی تکراری و سیاست‌گذاری تکراری(value iteration و policy iteration) چیست و در چه شرایطی ممکن است یکی را بر دیگری ترجیح دهیم؟
پاسخ: سیاست‌گذاری تکراری بر ارزیابی سیاست‌ها تمرکز دارد، در حالی که ارزش‌یابی تکراری به ارزیابی وضعیت‌ها یا جفت‌های وضعیت-عمل پرداخته و به طور ضمنی یک سیاست را استخراج می‌کند.
محدودیت‌های ارزش‌یابی تکراری عبارتند از:

1. هر تکرار زمان $O(|S|^2|A|)$ را می‌برد که می‌تواند پرهزینه باشد.
2. مقادیر بسیاری از وضعیت‌ها در یک تکرار تغییر نمی‌کند، اما فرآیند باید ادامه یابد اگر تغییری در برخی وضعیت‌ها رخ دهد.
3. گاهی سیاست مربوطه (که به‌عنوان V_k در نظر گرفته می‌شود) ممکن است به حد مطلوب رسیده باشد اما مقادیر همگرا نشده‌اند و این باعث می‌شود تکرارهای بی‌فایده ادامه یابد.
یکی از معایب سیاست‌گذاری تکراری(policy iteration) این است که هر تکرار نیاز به ارزیابی سیاست (policy evaluation) دارد که ممکن است یک فرآیند تکراری طولانی باشد. در برخی شرایط، سیاست‌گذاری تکراری (policy iteration) سریع‌تر از ارزش‌یابی تکراری (value iteration) همگرا می‌شود.

(ج) سیاست‌گذاری تکراری کی به پایان می‌رسد؟ بلافاصله پس از پایان (بدون محاسبات اضافی) آیا مقادیر سیاست بهینه را داریم؟
پاسخ: سیاست‌گذاری تکراری زمانی پایان می‌یابد که سیاست همگرا شود، یعنی وقتی $\pi_{new} = \pi_{old}$ پس از بهبود سیاست رخ دهد.
بعد از همگرایی، مقادیر ارزش‌های سیاست بهینه را داریم، چرا که ما ارزیابی سیاست را در آخرین تکرار انجام داده‌ایم.

(د) اگر در طی سیاست‌گذاری تکراری، فقط یک تکرار از به‌روزرسانی بلمن را به جای اجرای کامل تا همگرایی اجرا کنیم، چه تغییری رخ می‌دهد؟ آیا همچنان به سیاست بهینه می‌رسیم؟
پاسخ: بله، همچنان به سیاست بهینه دست می‌یابیم. این روش در واقع مشابه ارزش‌یابی تکراری می‌شود، زیرا شامل یک گام ارزیابی بر اساس بهترین سیاست فعلی است تا زمانی که سیاست همگرا شود.

سوال دوم) مسابقه

یک مثال تغییر یافته از مسابقه‌ی ربات خودرو را که در کلاس دیدیم در نظر بگیرید. در این بازی، خودرو به طور تصادفی تعدادی از فضاها را حرکت می‌کند که به‌طور مساوی احتمال دارد ۲، ۳ یا ۴ باشد. خودرو می‌تواند حرکت کند یا متوقف شود اگر مجموع فضاهای حرکت کرده کمتر از ۶ باشد.
اگر مجموع فضاهای حرکت کرده برابر یا بیشتر از ۶ باشد، بازی با پاداش ۰ به پایان می‌رسد. هنگامی که خودرو متوقف می‌شود، پاداش برابر با مجموع فضاهای حرکت کرده (تا حداکثر ۵) خواهد بود و بازی به پایان می‌رسد. برای عمل حرکت پاداشی وجود ندارد.
این مسئله را به‌عنوان یک MDP با وضعیت‌های {0, 2, 3, 4, 5, Done} فرمول‌بندی می‌کنیم.

(الف) تابع انتقال (transition function) برای این MDP چیست؟

پاسخ:

$$
\begin{aligned}
&T(s, Stop, Done) = 1, \text{ for } s \neq Done \\
&T(0, Move, s') = \tfrac{1}{3} \text{ for } s' \in \{2, 3, 4\} \\
&T(2, Move, s') = \tfrac{1}{3} \text{ for } s' \in \{4, 5, Done\} \\
&T(3, Move, 5) = \tfrac{1}{3} \\
&T(3, Move, Done) = \tfrac{2}{3} \\
&T(4, Move, Done) = 1 \\
&T(5, Move, Done) = 1 \\
&T(s, a, s') = 0 \text{ otherwise.}
\end{aligned}
$$

(ب) تابع پاداش برای این MDP چیست؟

پاسخ:

$$
\begin{aligned}
&R(s, Stop, Done) = s, s \leq 5 \\
&R(s, a, s') = 0 \text{ otherwise}
\end{aligned}
$$

(ج) ارزش‌یابی تکراری (valueiteration) برای ۴ تکرار با ۱ = ۷ را اجرا کنید.

پاسخ:

States	0	2	3	4	5
V_0	0	0	0	0	0
V_1	0	2	3	4	5
V_2	3	3	3	4	5
V_3	$\frac{10}{3}$	3	3	4	5
V_4	$\frac{10}{3}$	3	3	4	5

(د) سیاست بهینه چیست؟

پاسخ:

States	0	2	3	4	5
π^*	Move	Move	Stop	Stop	Stop

(ه) نتایج چگونه با 0.1 = γ تغییر می‌کند؟ دلیل آن را توضیح دهید.

پاسخ: با 0.1 = γ، بر پاداش‌های فوری تمرکز بیشتری داریم که باعث می‌شود الگوریتم حریص‌تر شود و پاداش‌های کوتاه‌مدت ارزش بیشتری نسبت به پاداش‌های بلندمدت پیدا کنند.

در این بازی، با ضریب تخفیف 0.1، فرآیند ارزش‌یابی تکراری در تعداد کمتری از تکرارها به همگرایی می‌رسد، اما به یک سیاست متفاوت می‌انجامد (حرکت، توقف، توقف، توقف، توقف). برای حالت ۲، الگوریتم به جای پاداش بلندمدت حرکت، ترجیح می‌دهد از پاداش کوتاه‌مدت توقف بهره ببرد.

(و) برای این MDP، دو iteration از تکرار سیاست (policy iteration) را برای یک step از این MDP اجرا کنید، با شروع از سیاست اولیه زیر و با استفاده از مقدار اولیه ۱ = γ. (π_2 را به همراه مراحل رسیدن به آن، تعیین کنید.)

$\pi_0 = Move, Stop, Move, Stop, Move$

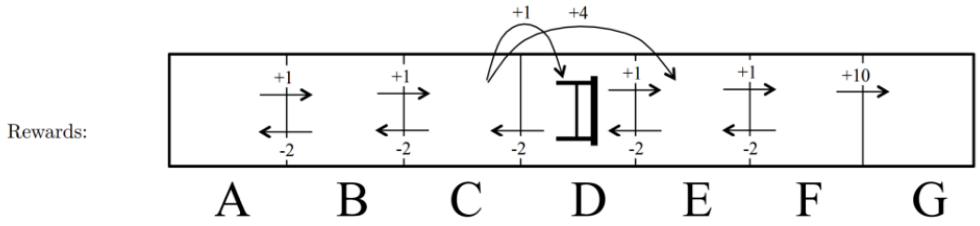
پاسخ:

States	0	2	3	4	5
π_0	Move	Stop	Move	Stop	Move
$V_0^{\pi_0}$	0	0	0	0	0
$V_1^{\pi_0}$	0	2	0	4	0
$V_2^{\pi_0}$	2	2	0	4	0
$V_3^{\pi_0}$	2	2	0	4	0
π_1	Move	Stop	Stop	Stop	Stop
$V_0^{\pi_1}$	0	0	0	0	0
$V_1^{\pi_1}$	0	2	3	4	5
$V_2^{\pi_1}$	3	2	3	4	5
$V_3^{\pi_1}$	3	2	3	4	5
π_2	Move	Move	Stop	Stop	Stop

نکته جانبی: در مرحله ارزیابی π_1 ما ارزیابی سیاست را با همه‌ی ارزش‌های صفر شروع کردیم؛ این روش را cold start می‌نامند. همچنین می‌توانستیم به جای آن، با ارزش‌های بهینه‌ای که در تکرار قبلی به دست آمده بودند ($V_3^{\pi_0}$) شروع کنیم که معمولاً سریع‌تر به همگرایی می‌رسد.

سوال سوم) دوی با مانع

در نظر بگیرید که یک MDP داریم که یک مسیر دویدن از روی موانع را مطابق شکل زیر نشان می‌دهد. یک مانع در مربع D و وضعیت پایانی در مربع G وجود دارد. عامل می‌تواند به سمت چپ یا راست بدود. اگر عامل در مربع C باشد، می‌تواند به سمت راست بدود ولی به جای آن می‌تواند بپرد، که این عمل ممکن است منجر به سقوط به مربع مانع D شود. پاداش‌ها در زیر نمایش داده شده‌اند و ضریب تخفیف را با مقدار 1 = γ فرض کنید.



اکشن‌ها:

- راست: به طور قطعی به راست حرکت می‌کند. (در خانه C قابل اتخاذ نیست.)
- چپ: به طور قطعی به چپ حرکت می‌کند.
- پرش: به طور تصادفی به راست می‌پرد و فقط برای خانه C قابل اتخاذ است. احتمال موفقیت پرش برابر با 50٪ است.

الف) برای سیاست π که همیشه حرکت مستقیم را پیشنهاد می‌دهد (همیشه راست یا پرش)، مقدار $V^{\pi}(C)$ را محاسبه کنید.

پاسخ: در حالتی که پرش موفقیت‌آمیز باشد (با احتمال ۵۰٪)، جمع پاداش‌ها از C برابر ۱۵ = ۴ + ۱۰ + ۱ و اگر با شکست مواجه شود برابر ۱۳ = ۱۰ + ۱ + ۱ + ۱ خواهد بود. پس:

$$V^{\pi}(C) = \frac{15+13}{2} = 14$$

ب) دو بار پیمایش ارزش (value iteration) را انجام دهید و مقادیر زیر را حساب کنید. مقداردهی اولیه همه ارزش‌ها برابر صفر است.

- $V_2(B)$
- $Q_2(B, Right)$
- $Q_2(B, Left)$

پاسخ: به یاد داشته باشید که V_{k+1} و Q_{k+1} از مقادیر V_k و Q_k برای حالات پسین استفاده می‌کنند.
$V_2(B)$ ماکسیمم Qها برای B است.

نکته: دو پیمایش برای value iteration مقدار را برای یک h = 2 قدم در افق اپیزود محاسبه می‌کند.
بنابراین $V_2(s)$برای هر وضعیت s برابر ماکسیمم خروجی مورد انتظار پس از دو قدم بعدی می‌باشد.

ج) برای خانه‌های خالی جدول زیر، مقادیر Q-valueها را با بروزرسانی‌هایی که از اعمال انتقال مشخص شده برای Q-learning به دست می‌آیند، پر کنید. از نرخ یادگیری α = 0.5 استفاده کنید و فرض کنید همه Q-valueها در ابتدا برابر صفر بودند. خانه‌هایی که تغییری نمی‌کنند خالی بگذارید.

Episode

s	a	r	s	a	r	s	a	r	s	a	r	s
C	*jump*	+4	E	*right*	+1	F	*left*	-2	E	*right*	+1	F

	$Q(\mathrm{C}, left)$	$Q(\mathrm{C}, jump)$	$Q(\mathrm{E}, left)$	$Q(\mathrm{E}, right)$	$Q(\mathrm{F}, left)$	$Q(\mathrm{F}, right)$
Initial	0	0	0	0	0	0
Transition 1						
Transition 2						
Transition 3						
Transition 4						

پاسخ: جدول مقادیر Q-valueها در هر مرحله از انتقال، با استفاده از به‌روزرسانی‌های حاصل از Q-learning و با نرخ یادگیری α = 0.5، به صورت زیر پر شده است. مقادیر اولیه برای همه Q-valueها برابر صفر بودند.

	$Q(\mathrm{C}, left)$	$Q(\mathrm{C}, jump)$	$Q(\mathrm{E}, left)$	$Q(\mathrm{E}, right)$	$Q(\mathrm{F}, left)$	$Q(\mathrm{F}, right)$
Initial	0	0	0	0	0	0
Transition 1		2				
Transition 2				0.5		
Transition 3					−0.75	
Transition 4				0.75		

Q-learning update:

$Q'(s,a) = Q(s,a) + \alpha[r + max_{a'}\gamma Q(s',a') - Q(s,a)$

$2 = 0 + 0.5(4 + 0 - 0)$

$0.5 = 0 + 0.5(1 + 0 - 0)$

$-0.75 = 0 + 0.5(-2 + .0.5 - 0)$

$0.75 = 0.5 + 0.5(1 + 0 - 0.5)$

Logic

سوال اول) فرزندان محمد

وقتی از محمد درباره سن فرزندانش پرسیدند، او گفت: «آلیس کوچکترین فرزند من است، به شرطی که بیل کوچکترین نباشد. همچنین آلیس کوچکترین فرزند من نیست، اگر کارل کوچکترین نباشد.» دانش پایه‌ای برای توصیف این مسئله و این واقعیت که فقط یکی از این سه فرزند می‌تواند کوچکترین باشد را بنویسید. سپس با استفاده از روش حل (resolution)، نشان دهید که بیل کوچکترین فرزند اوست.

پاسخ: فرض کنیم A, B, C به ترتیب بیانگر این باشند که آلیس، بیل، و کارل کوچکترین فرزندان هستند. به این ترتیب قواعد زیر را برای دانش پایه داریم:

1. $A \vee B \vee C$ (فقط یکی از بچه‌ها باید کوچکترین باشد.)

2. $A \vee \neg B$ (آلیس و بیل نمی‌توانند هر دو کوچکترین باشند.)

3. $A \vee \neg C$ (آلیس و کارل نیز نمی‌توانند هر دو کوچکترین باشند.)

4. $B \vee \neg C$ (بیل و کارل نیز نمی‌توانند همزمان کوچکترین باشند.)

اطلاعاتی که محمد ارائه داده به صورت زیر قابل بیان است:

5. $A \vee B$ (اگر بیل کوچکترین نباشد، آلیس کوچکترین است. یعنی $\neg A \Rightarrow B$.)

6. $\neg A \vee C$ (اگر کارل کوچکترین نباشد، آلیس کوچکترین نیست. یعنی $\neg A \Rightarrow C$.)

برای نشان دادن اینکه بیل کوچکترین است، فرض می‌کنیم که بیل کوچکترین نیست:

7. $\neg B$ (فرض کنیم بیل کوچکترین نیست.)

با استفاده از روش حل (resolution) به نتیجه زیر می‌رسیم:

8. از (5, 7): A

9. از (3, 6): $\neg A$

10. از (8, 9): $\bot$

سوال سوم) مجله معمایی

در انتهای یک مجله معمایی را می‌بینید: «فرض کنید دروغ‌گوها همیشه چیزی را که غلط است می‌گویند و راست‌گوها همیشه حقیقت را می‌گویند. همچنین فرض کنید که امین یا دروغ‌گو است یا راست‌گو.» این معما سپس حقایق دیگری را درباره امین ارائه می‌دهد و می‌پرسد که آیا امین باید راست‌گو باشد؟ شما این حقایق را به منطق گزاره‌ای تبدیل کرده و یک روش حل را بر روی رایانه اجرا می‌کنید. از آنجایی که اشتباهی مرتکب نمی‌شوید، رایانه پاسخ صحیح را به شما می‌دهد. شما از رایانه می‌پرسید که آیا حقایق به این نتیجه می‌رسند که امین راست‌گو است.

الف) رایانه به شما می‌گوید که حقایق به این نتیجه می‌رسند که امین راست‌گو است. از آنجا که متن بیان کرده که امین یا دروغ‌گو است یا راست‌گو، آیا می‌توانید نتیجه بگیرید که امین دروغ‌گو نیست؟

پاسخ: بله، از آنجا که می‌دانیم امین باید راست‌گو باشد (اگر حقایق درست باشند)، او نمی‌تواند دروغ‌گو باشد.

ب) رایانه به شما می‌گوید که حقایق به این نتیجه نمی‌رسند که امین راست‌گو است. از آنجا که متن بیان کرده که امین یا دروغ‌گو است یا راست‌گو، آیا می‌توانید نتیجه بگیرید که امین دروغ‌گو است؟

پاسخ: خیر، ممکن است امین راست‌گو نباشد و دروغ‌گو باشد (اگر حقایق درست باشند)، اما شما نمی‌توانید با قطعیت این را بگویید. ممکن است رایانه اطلاعات کافی برای نتیجه‌گیری نداشته باشد. اگر می‌خواهید بدانید که آیا امین باید دروغ‌گو باشد، باید از رایانه بپرسید که آیا حقایق منجر به این می‌شوند که امین دروغ‌گو باشد.

سوال پنجم) FOL (صداهای بوق الکترونیکی)

گزاره‌های زیر را در نظر بگیرید که در آن دو گزاره به زبان گفتاری و دو گزاره به صورت منطق مرتبه اول[1] ارائه شده است.

1) همه ربات‌های کت‌بوت (CatBot robots) در شب صداهای بوق الکترونیکی تولید می‌کنند.

2) $\forall x \; \forall y \; (\text{Have}(x, y) \wedge \text{Real\_Cat}(y) \Rightarrow \sim \exists z \; (\text{Have}(x, z) \Rightarrow \text{Mice}(z)))$.

3) افراد سبک‌خواب (Light sleepers) هیچ چیزی که در شب صداهای بوق الکترونیکی تولید کند، ندارند.

4) سوزی (Susie) یا یک گربه واقعی (Real Cat) یا یک ربات کت‌بوت (CatBot robot) دارد.

[1] First Order Logic

نتیجه-) Light_Sleeper(Susie) ⇒ ~∃z (Have(Susie, z) ∧ Mice(z)).
5

الف) گزاره‌های 1، 3 و 4 را به صورت فرمول‌های خوش‌ساختار[2] در منطق مرتبه اول با استفاده از گزاره‌های زیر بنویسید:

- CatBot_Robot(x)

- Have(x, y)

- Make_Noise(x)

- Real_Cat(x)

- Light_Sleeper(x)

سپس گزاره 2 و نتیجه-5 را به زبان فارسی بنویسید.

پاسخ: به ترتیب:

1. ∀x CatBot_Robot(x) ⇒ Make_Noise(x)
2. هر کسی که گربه واقعی دارد، هیچ موشی ندارد.
3. ∀x (Light_Sleeper(x) ⇒ ∀y (Have(x, y) ⇒ ¬Make_Noise(y)))
4. ∃x (Have(Susie, x) ∧ (Real_Cat(x) ∨ CatBot_Robot(x)))
5. اگر سوزی سبک‌خواب باشد، آنگاه سوزی هیچ موشی ندارد.

(ب) هر فرمول خوش‌ساختار را با معرفی ثابت‌ها به جای کوانتورهای وجودی (اسکولم‌سازی ساده)، و بازنویسی همه گزاره‌ها به صورت CNF تبدیل کنید.

پاسخ: قوانین در CNF:

1. از گزاره 1:

¬CatBot_Robot(x) ∨ Make_Noise(x)

2. از گزاره 2 (اسکولم‌سازی و تبدیل به CNF):

[2] Well-Formed Formula

¬Have(x, y) ∨ ¬Real_Cat(y) ∨ (¬Have(x, z) ∨ ¬Mice(z))

3. از گزاره 3:

¬Light_Sleeper(x) ∨ ¬Have(x, y) ∨ ¬Make_Noise(y)

4. از گزاره 4 (اسکولم‌سازی):

4a) Have(Susie, c1) 4b) Real_Cat(c1) ∨ CatBot_Robot(c1)

5. نفی نتیجه (برای اثبات با رزولوشن):

5a) ¬Light_Sleeper(Susie) 5b) Have(Susie, c2) 5c) Mice(c2)

(ج) نتیجه را با استفاده از رزولوشن اثبات کنید. در این مرحله، باید پنج گزاره به صورت CNF به عنوان عبارات پایگاه دانش، و سه گزاره CNF به عنوان نتیجه داشته باشید. لطفاً هنگام استفاده از قانون رزولوشن در اثبات خود، به شماره گزاره‌ها اشاره کنید و هنگام پیشروی در اثبات خود، گزاره‌های جدیدی که به دست می‌آورید را شماره‌گذاری کنید.

پاسخ:

6. رزولوشن بین گزاره‌های 1 و 4b:

Real_Cat(c1) ∨ Make_Noise(c1)

7. رزولوشن بین گزاره‌های 2 و 5c:

¬Have(x, y) ∨ ¬Real_Cat(y) ∨ ¬Have(x, z)

8. رزولوشن بین گزاره 7 و 5b: (با جایگزینی x = Susie)

¬Have(Susie, y) ∨ ¬Real_Cat(y)

9. رزولوشن بین گزاره‌های 6 و 8:

¬Have(Susie, c1) ∨ Make_Noise(c1)

10. رزولوشن بین گزاره‌های 9 و 4a:

Make_Noise(c1)

11. رزولوشن بین گزاره‌های 3 و 10: (با جایگزینی x = Susie و y = c1)

¬Light_Sleeper(Susie) ∨ ¬Have(Susie, c1)

12. رزولوشن بین گزاره‌های 11 و 4a:

¬Light_Sleeper(Susie)

13. رزولوشن بین گزاره‌های 12 و 5a: -> تناقض (False)

از آنجا که به تناقض رسیدیم، نتیجه اثبات شد.